Reducing multi-qubit interactions in adiabatic quantum computation. Part 2: The "split-reduc" method and its application to quantum determination of Ramsey numbers

Emile Okada[1,*]
[1]*Department of Mathematics, Cambridge University, CB2 3AP, Cambridge, UK.*

Richard Tanburn[2,†]
[2]*Mathematical Institute, Oxford University, OX2 6GG, Oxford, UK.*

Nikesh S. Dattani[3,4,‡]
[3]*School of Materials Science and Engineering, Nanyang Technological University, 639798, Singapore, and*
[4]*Fukui Institute for Fundamental Chemistry, 606-8103, Kyoto, Japan*

Quantum annealing has recently been used to determine the Ramsey numbers $R(m,2)$ for $4 \leq m \leq 8$ and $R(3,3)$ [Bian *et al.* (2013) PRL **111**, 130505]. This was greatly celebrated as the largest experimental implementation of an adiabatic evolution algorithm to that date. However, in that computation, more than 66% of the qubits used were auxiliary qubits, so the sizes of the Ramsey number Hamiltonians used were tremendously smaller than the full 128-qubit capacity of the device used. The reason these auxiliary qubits were needed was because the best quantum annealing devices at the time (and still now) cannot implement multi-qubit interactions beyond 2-qubit interactions, and they are also limited in their capacity for 2-qubit interactions. We present a method which allows the full qubit capacity of a quantum annealing device to be used, by reducing multi-qubit and 2-qubit interactions. With our method, the device used in the 2013 Ramsey number quantum computation could have determined $R(16,2)$ and $R(4,3)$ with under 10 minutes of runtime.

I. INTRODUCTION

The capacities and limits for adiabatic quantum computers (AQCs) to outperform classical computers, and to speed-up the solution to discrete optimization problems has recently been discussed in [1]. As discussed in [1], the quantum annealing devices with the largest qubit capacities tend only to allow up to at most 2-qubit interactions, and are even limited in the 2-qubit interactions allowed. Similarly, even when solving a discrete optimization problem on a classical computer, high-order terms rapidly make the problem more difficult. If only up to linear terms (1 qubit terms) are present in the Hamiltonian (objective function), then finding the solution to the problem is trivial, but if quadratic terms (2-qubit terms) are allowed the problem becomes NP complete.

Nevertheless, an enormous body of work has been done on efficient algorithms for quadratic unconstrained Boolean optimization (QUBO) problems, and it is known that if all coefficients of quadratic terms are negative, the solution can be found in polynomial time [2–5]. When cubic (3-qubit) terms and beyond are present, another leap in difficulty arises, and most of the effort is typically spent on quadratizing such objective functions (Hamiltonians). Most quadratization techniques work by adding auxiliary variables (qubits), and while algorithms for finding solutions to discrete optimization problems often scale exponentially with the number of variables, it is *still* often desirable to remove cubic terms and higher at the expense of adding more variables.

However, quantum annealing devices to date are very limited in qubit capacity (the largest device reported to date having only about 2 kiloqubits or 258 qubytes). Therefore, adding auxiliary qubits is usually not an option if any benefit over traditional computation methods is desired for any relevant problem. In Part 1 [1] we demonstrated a method called "deduc-reduc" which reduces multi-qubit interactions without adding auxiliary qubits, and for the integer factorization problem, managed to eliminate thousands of 4-qubit and 3-qubit interactions with just a few seconds of CPU time. A drawback of this method is that some deduction must be made which relates the variables of the discrete optimization problem (an example of such a deduction could be $x_1 + x_2 = 1$). Such deductions arise naturally for the problem of integer factorization, but there is no reason to believe that such deductions can be made for an arbitrary discrete optimization problem.

In this paper we present a method for reducing multi-qubit interactions without adding auxiliary qubits *and* without the need for any deductions, but it increases the number of objective functions that need to be minimized to find the solution to the original objective function, and adding auxiliary qubits improves the method. We call this method "split-reduc" since it iteratively *splits* the Hamiltonian into separate Hamiltonians in order to *reduce* multi-qubit terms. We give very conservative lower and upper bounds on the number of new objective functions created, and we showcase split-reduc on

* eto25@cam.ac.uk
† richard.tanburn@hertford.ox.ac.uk
‡ nike.dattani@gmail.com

the Hamiltonian used in the determination of Ramsey numbers using quantum annealing, as in [6].

II. A QUICK EXAMPLE

Let us demonstrate the method with the simple objective function $H = 1+x_1x_2x_5+x_1x_6x_7x_8+x_3x_4x_8-x_1x_3x_4$ and an adiabatic quantum computer (AQC) that only has 8 qubits and only allows up to at most 2-qubit interactions. Due to the restriction on the number of qubits we cannot reduce the qubic terms to quadratic terms by introducing auxiliary variables. The simple, but effective idea is then to "split" the objective function into two by setting a variable to its two possible values (0 or 1). In this case x_1 is the obvious choice to split over since it is present in the most terms and contributes to the quartic term. Setting x_1 to 0 results in the objective function $H_0 = 1 + x_3x_4x_8$ and setting x_1 to 1 results in $H_1 = 1 + x_2x_5 + x_6x_7x_8 + x_3x_4x_8 - x_3x_4$.

H_0 still contains cubic terms so we have the choice to split H_0 further. However at this point we have 5 unused qubits so we could also stop here by using one of them as an auxiliary variable to quadratize the objective function. H_1 on the other hand is however still a bit too complicated for our quantum computer to handle. It contains cubic terms and requires 7 qubits out of the 8 qubit capacity of the AQC, so we split again, this time over x_8. We get the objective functions $H_{10} = 1+x_2x_5+x_6x_7$ and $H_{11} = 1+x_2x_5+x_3x_4$. Both only contain quadratic terms so we have succeeded in turning our Hamiltonian into 3 separate Hamiltonians that can each be implemented on the AQC. In general, we can reduce the number of splits necessary, by combining this approach with established methods for quadratization techniques that introduce auxiliary variables.

III. THE METHOD

We now demonstrate the method in full generality. We first define two cost functions:

1. $C(H)$ tells us whether or not we need to split the Hamiltonian any further, and
2. $C_H(x_i)$ tells us which variable to choose for the splitting at each step, by assigning a cost to each variable.

The idea is that we keep splitting the Hamiltonian, according to the variable selected from $C_H(x_i)$, until $C(H)$ is true.

A. Choosing $C(H)$

Different problems may involve different constraints. If a device can only handle 2-qubit interactions (such as SQUID-based quantum annealers as in [7]) we might want a different $C(H)$ than if the device can handle 3-qubit interactions (such as NMR-based AQCs as in [8]). If we cannot, or do not want to add any auxiliary variables, then we do not need the function $C(H)$.

If we wish to allow the addition of auxiliary variables, then for each term t in H, we determine how many auxiliary variables $n_{\text{aux},t}$ will be needed in order to reduce t to our desired order (quadratic order for a device that allows 2-qubit interactions, cubic order for a device that allows 3-qubit interactions, etc.). The function is then

$$C(H) = n + \sum_t n_{\text{aux},t} \leq Q, \tag{1}$$

where n is the original number of qubits before any auxiliary qubits were added and Q is our AQC's qubit capacity.

Since the most successful quantum annealing experiments performed thus far have been on architectures which do not allow higher than 2-qubit interactions, we will give an example of how to choose $C(H)$ for such a device. There are many different ways to quadratize a term t, and each of these methods will have its own $n_{\text{aux},t}$, but we know from [6] that $n_{\text{aux},t}$ will not be more than

$$n_{\text{aux},t} = \mathcal{R}\left(\text{order}(t) - 2\right), \tag{2}$$

where $\mathcal{R}$ is the Ramp function (see Appendix for details about the quadratization method which only needs at most this many auxiliary variables). For terms that are already quadratic, linear, or constant, $\text{order}(t) \leq 2$ so $\mathcal{R}\left(\text{order}(t) - 2\right) = 0$ and no auxiliary variables are necessary. If t is, for example, quintic, then $\mathcal{R}\left(\text{order}(t) - 2\right) = \mathcal{R}\left(5 - 2\right) = 3$ so the *maximum* number of auxiliary qubits added to the cost function in Eq. 1 is 3.

Therefore, if our goal is to quadratize the Hamiltonian for a device that only allows up to 2-qubit interactions, and we are limited to only Q total qubits, then the cost relation is

$$C(H) = n + \sum_t \mathcal{R}\left(\text{order}(t) - 2\right) \leq Q. \tag{3}$$

B. Choosing $C_H(x_i)$

As in the previous section, our choice of $C_H(x_i)$ depends on the situation. We may wish to only have quadratic terms without introducing *any* auxiliary variables, or we may want to choose a cost function that picks the variable that appears most frequently in the undesired (super-quadratic) terms. If we choose Eq. 3 to be our cost formula, we may wish to choose a greedy $C_H(x_i)$ that simply minimizes the number of auxiliary variables that would need to be added in order

to quadratize it. In conjunction with the cost formula in Eq. 3, we may define:

$$C_H(x_i) = \sum_t \left[I_{x_i,t} \cdot \mathcal{R}\left(\text{order}(t) - 2 + 1\right)\right], \tag{4}$$

where $I_{x_i,t}$ is 1 if x_i appears in t and 0 otherwise. The indicator function makes sure we only count terms in which x_i appears, and $\mathcal{R}\left(\text{order}(t) - 2\right)$ is of course the maximum number of auxiliary variables needed to quadratize term t, but we include $+1$ to account for when the variable is set to 1. For the splitting, we then choose the variable x_i with the biggest $C_H(x_i)$.

IV. ESTIMATES ON THE NUMBER OF SPLITTINGS

In Section II the benefits of splitting were clear. We only needed 3 objective functions in the end, which is a small fraction of the search space of size 2^8. But what about in general? It is not difficult to construct cases in which the number of splits required blows up. However, this is often not the case. We may think of the splitting process as giving rise to a binary tree. The root of the tree is the original objective function and each node has two branches or zero branches (from splitting or not splitting respectively). Establishing tight analytic bounds on the number of leaves may seem tricky, but with simple assumptions, we show that we can estimate upper and lower bounds remarkably well.

A. Heuristic bounds

Let us start with the most basic lower bound we can imagine. We can assume that the shortest path from the root of the tree to a Hamiltonian that satisfies all hardware requirements is found by successively choosing the variable with the highest cost and setting it to 0. While false in cases like $H = (1-x_1)(1-x_2)(1-x_3)$ where setting any variable to 1 is preferable to setting it to 0, it is usually true when a lot of the monomials have the same sign or many terms do not share variables. Likewise, we can assume that the longest path is found by setting the highest cost variable to 1 at each split.

Provided the above conditions hold, finding the lengths of the extreme paths then becomes trivial and requires at most n substitutions. Once we know these lengths, call them l and s for the longest and shortest path respectively, we know a lower bound is 2^s and an upper bound is 2^l.

B. More sophisticated estimates based on combinatorics

The above bounds are not very tight, so we formulate a more sophisticated estimate, and we ensure that the method tends to overestimate the number of splits. Let us make the stronger assumption that if s variables were set to 0 to obtain the shortest path, then setting s variables to 0 will always be sufficient to obtain a Hamiltonian that satisfies the hardware requirements (a "desirable Hamiltonian"). The reason this tends to overestimate (and hence could be considered an upper bound) is that it ignores the fact that setting a variable to 1 also helps simplify the Hamiltonian. Using the same number of operations as before, we can now find better bounds. We know that either s variables are set to 0 to obtain a desirable Hamiltonian, or l variables have been set to 0 or 1 (since l is the length of the longest path). To count the number of such paths consider an l-bit string

$$x_1 x_2 \ldots x_l. \tag{5}$$

There are $\binom{l}{s}$ ways to choose at which stage the s variables are set to 0. Filling in all the blank spaces before the last 0 with 1's and leaving the rest empty characterizes all desirable Hamiltonians in which s variables were set to 0. If k variables are set to 0 where $k < s$, then there are $\binom{l}{k}$ desirable Hamiltonians since all we need is that l variables have been given a value. Thus, the number of Hamiltonians is

$$\sum_{k=0}^{s} \binom{l}{k}. \tag{6}$$

What happens if we do not ignore the reducing potential of setting a variable to 1? Suppose R_i right moves (setting variables to 1) reduces a Hamiltonian as much as R_i left moves (setting variables to 0). We want to count the number of paths where k left moves are made. Since $k < s$, left moves alone will not simplify our Hamiltonian. We will need R_{s-k} right moves to make up the difference. However we can include a full $R_{s-k+1} - 1$ right moves since without that last right move the Hamiltonian will not be desirable. That means we have $k + R_{s-k+1} - 1$ slots to fill with k left moves and $R_{s-k+1} - 1$ right moves. The number of such paths is simply $\binom{R_{s-k+1}-1+k}{k}$ and so the total number of desirable Hamiltonians is

$$1 + \sum_{k=1}^{s} \binom{R_{s-k+1} - 1 + k}{k}. \tag{7}$$

This too is likely a slight over-estimate since in practice we often encounter Hamiltonians that don't require exactly R_i right moves, but rather some number in the neighborhood of R_i. Now only one issue remains: calculating R_i. The authors of this paper prefer overestimates to bad estimates, so we shall try to find R_i that are likely larger than they need to be. We start by successively making right moves until a desirable Hamiltonian is reached. Before each right move, however, we note how many left moves would be needed to reach a

Table I. Performance of split-reduc on R(4,3)

Number of vertices	Total size of search space	# of Hamiltonians needed with $Q = 128$	Upper bound from Section IV B
6	2^{15}	1	1
7	2^{21}	9	9
8	2^{28}	169	187
9	2^{36}	6 716	9 097
Number of vertices	Total size of search space	# of Hamiltonians needed with $Q = 50$	Upper bound from Section IV B
6	2^{15}	9	9
7	2^{21}	126	156
8	2^{28}	3 367	3 893
9	2^{36}	177 754	346 758
Number of vertices	Total size of search space	# of Hamiltonians needed with $Q = 30$	Upper bound from Section IV B
6	2^{15}	24	27
7	2^{21}	398	573
8	2^{28}	13 389	22 246
9	2^{36}	829 055	1 932 743

desirable Hamiltonian from this point and thus generate a sequence of length $l + 1$ (the number of nodes on the path). If the sequence is non-increasing, this method is likely to produce a good estimate since it conforms to the assumptions we made. We then define R_i to be the position of the last occurrence of $s - i + 1$ in the sequence since that is the point at which adding a right move would remove the need for a left move. If the sequence is not non-increasing then this will just produce a higher upper bound and if the sequence skips a number (by for example, decreasing by two), we define R_i instead to be the last occurrence of a number larger than $s - i + 1$. This procedure involves $\mathcal{O}(n^2)$ steps since the max length of any path is n.

To demonstrate the idea, we consider the objective function from Section II. The shortest path is $s = 1$ and the longest is $l = 2$ (see Fig. 1). The sequence generated by the above procedure is $(1, 1, 0)$ so $R_1 = 2$. That means the number of splits is $1 + \binom{2-1+1}{1} = 3$, which happens to be correct!

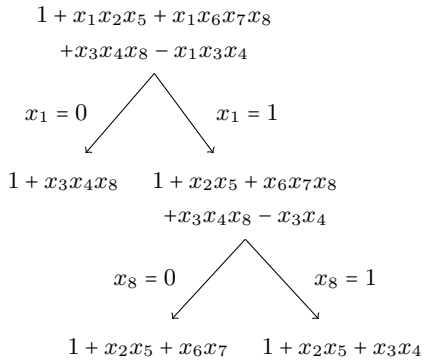

Figure 1. A tree representation of the splitting of f.

PERFORMANCE ON RAMSEY NUMBER HAMILTONIANS

It has been shown in [6, 9] that finding the Ramsey number $R(m, n)$ is equivalent to finding what number of vertices is needed for the ground state of a certain Hamiltonian to have an energy of greater than 0. For each (m, n) a Hamiltonian is made to be associated with a graph G with N vertices, and counts the number of complete subgraphs K_m, and n-independent sets. The first number N such that the global minimum of $H(m, n, N)$ is not 0, is defined as the Ramsey number $R(m, n)$.

C. $R(m, 2)$

The largest Ramsey number determined by the quantum annealing device in [6] was $R(8, 2) = 8$. The Hamiltonian for this case is:

$$H = \sum_{k=1}^{L_m} 1 - a_k + \prod_{k=1}^{L_m} a_k, \quad L_m = \binom{m}{2} = 28 \tag{8}$$

and it is clear we need to deal with a 28 qubit interaction, because the second term is a product of 28 qubits. In [6] they introduce auxiliary variables. We will use split-reduc instead.

Due to the complete symmetry of all the variables we can pick one at random at each step and split it. If we choose not to allow auxiliary variables, and aim to split H until it is quadratic, we end up with the following Hamiltonians after splitting:

$$1 + \sum_{k=1+i}^{28} 1 - a_k \text{ for } 1 \le i \le 26 \text{ and } 2 - a_{27} - a_{28} + a_{27}a_{28}. \tag{9}$$

If we had used Eq. 7 to predict the number of objective functions we would have found that $s = 1$, and $l = R_1 = 26$. That would mean the number of Hamiltonians is $1 + \binom{R_1}{1} = 1 + 26 = 27$, which also happens to be correct!

For $R(m,2)$ in general the combinatorial estimates in Section IV B are provably correct, and $s = 1$, $l = R_1 = \binom{m}{2} - 2$. Thus with the 128 qubits available in the quantum annealing device of [6], the authors could also have calculated $R(16,2)$ with *at most* $\binom{16}{2} - 2 = 118$ runs. Section F of the Supplementary Information of [6] explains that the annealing runtimes tend to be around 2.5 ms, which includes a 1.5 ms delay for reading out the answer from the machine. Therefore, 118 runs on the device is feasible within 1 second. It is clear that the quantum annealing device of [6] is not so much runtime limited for this problem, as it is limited by the qubit capacity. We also note that 118 runs on the quantum annealing device is 35 orders of magnitude smaller than the size of the total search space if a brute force search were to be attempted to find $R(16,2)$.

Furthermore, while $R(16,2)$ was the largest $R(m,2)$ Ramsey number that could have been determined by the 128 qubit device used in [6] had they used split-reduc, we note that the newest version of that device has a qubit capacity of $Q = 2048$, meaning that we could now determine $R(64,2)$ which requires $\binom{64}{2} = 2016$ qubits, and would only require *at most* $\binom{64}{2} - 2 = 2014$ runs on the device.

D. $R(m,3)$

The $R(m,2)$ numbers have very simple objective functions. $R(m,3)$ Ramsey numbers are much more complicated, and the only one that was found in [6] was $R(3,3)$. The Hamiltonian for $R(4,3)$ for each N is too lengthy to present here, but can be derived from [6] and has at most 6-qubit interactions. Therefore, we apply split-reduc with Eq. 3 as our choice of $C(H)$ and Eq. 4 as our choice of C_H. Table I shows how close our overestimates from Section IV B are for $R(4,3)$, where we know that the required number of vertices (and hence the Ramsey number itself) is 9. While minimizing 6716 Hamiltonians would only take a few seconds on the quantum annealing device of [6], we note that this device has *another* restriction, which was not relevant for $R(m,2)$ because every term after split-reduction was linear at most (except for the last one). While the split-reduced $R(4,3)$ Hamiltonians meet the requirement that they are all quadratic at most, the quantum device of [6] also requires that the quadratic couplings can be implemented on their "chimera" graph.

In their example, $R(8,2)$ with $N = 8$ could be determined with a Hamiltonian that after quadratization had 54 qubits, and required 30 more qubits to chimerize the connectivity of the 54-vertex graph describing the connections between all qubits in the Hamiltonian. Therefore, for $R(4,3)$, if we choose the case in Table I that uses at most $Q = 50$ qubits in the split-reduced Hamiltonians, it is reasonable to assume that each of the resulting 177 754 Hamiltonians could be chimerized using the 72 qubits remaining in the 128-qubit device. Once again, if each minimization again took 2.5 ms, $R(4,3)$ would be determined within 10 minutes.

V. CONCLUSION

This is the second paper of a 2-part series on techniques for reducing multi-variable terms in discrete optimization problems. The first method is called "deduc-reduc" because it uses *deductions* to *reduce* the multi-qubit (multi-variable) terms in the Hamiltonian (objective function), and is presented in [1] with an application to the quantum factorization of numbers larger than 56153, which is currently the largest number factored on a quantum device [10]. Deduc-reduc can also be used to reduce multi-qubit interactions in the Ramsey number Hamiltonians discussed in the present paper, but we wished to focus only on the split-reduc method in this paper. Combining deduc-reduc, split-reduc, and a third algorithm we have recently devised for reducing the size of the search space for the Ramsey number discrete optimization problem, we are able to establish estimated runtimes for some of the presently undetermined Ramsey numbers such as $R(6,4)$, and $R(10,3)$ [11].

APPENDIX: QUADRATIZATION METHOD NEEDING AT MOST $\mathcal{R}\,(\mathrm{order}(t) - 2)$ AUXILIARY QUBITS

One way to quadratize a high-order term is to use the penalty function presented in Section II of the Supplementary Information of [6]:

$$P(a_1, a_2; b) = a_1a_2 - 2(a_1 + a_2)b + 3b, \tag{10}$$

which obtains a minimum of 0 only if $b = a_1 a_2$. Therefore if our Hamiltonian has a high-order term such as:

$$a_1 a_2 a_3 \dots a_n, \tag{11}$$

we can reduce its order by one, by replacing $a_1 a_2$ with a new variable b:

$$a_1 a_2 a_3 \dots a_n \to b a_3 \dots a_n + \lambda P(a_1, a_2; b), \tag{12}$$

for a scalar λ that is sufficiently large to not introduce any spurious minima (this is the "deduc-reduc" method of Part 1 of this paper [1], with $b = a_1 a_2$ as the deduction, and the choice of λ is discussed there). By construction, whether the LHS or RHS of Eq. 12 is considered, the unique minimum/minima will be the same, but the LHS has order n and the RHS does not have any terms greater than order $n - 1$.

Our reduced term

$$b a_3 \dots a_n \tag{13}$$

can then be further reduced by choosing another 2 variables to transform. Repeatedly applying this method allows us to quadratize a term t with *at most* $\mathrm{order}(t) - 2$ applications, which explains Eq. 2 in the main text.

ACKNOWLEDGMENTS

We gratefully thank Oliver Lunt of Oxford University's Trinity College for careful proofreading of the manuscript.

[1] R. Tanburn, E. Okada, and N. Dattani, Physical Review A (submitted) (2015), arXiv:1508.04816.
[2] G. Nemhauser and L. Wolsey, *North-Holland Mathematics Studies*, North-Holland Mathematics Studies, Vol. 59 (Elsevier, 1981) pp. 279–301.
[3] J. B. Orlin, Mathematical Programming **118**, 237 (2007).
[4] S. Jegelka, H. Lin, and J. A. Bilmes, in *Advances in Neural Information Processing Systems* (2011) pp. 460–468.
[5] E. Boros and P. L. Hammer, Discrete Applied Mathematics **123**, 155 (2002).
[6] Z. Bian, F. Chudak, W. G. Macready, L. Clark, and F. Gaitan, Physical Review Letters **111**, 130505 (2013).
[7] T. F. Ronnow, Z. Wang, J. Job, S. Boixo, S. V. Isakov, D. Wecker, J. M. Martinis, D. A. Lidar, and M. Troyer, Science **345**, 420 (2014).
[8] N. Xu, J. Zhu, D. Lu, X. Zhou, X. Peng, and J. Du, Physical Review Letters **108**, 130501 (2012).
[9] F. Gaitan and L. Clark, Physical Review Letters **108**, 010501 (2012).
[10] N. S. Dattani and N. Bryans, Physical Review Letters (revisions requested) (2015), arXiv:1411.6758.
[11] E. Okada, R. Tanburn, and N. S. Dattani, Physical Review A (in preparation) (2015).